
Printed by EAST

UserID: gdistefano

Computer: WS10764

Date: 02/25/2008

Time: 11:38

Document Listing

| Document | Image pages | Text pages | Error pages |
|---------------|-------------|------------|-------------|
| JP 2005100308 | 14 | 0 | 0 |
| Total | 14 | 0 | 0 |

(19) 日本国特許庁 (JP)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開2005-100308

(P2005-100308A)

(43) 公開日 平成17年4月14日 (2005.4.14)

| | | |
|----------------------------|-------------------|-------------|
| (51) Int. Cl. ⁷ | F I | テーマコード (参考) |
| G06 F 3/14 | G06 F 3/14 350 A | 5 B 0 6 9 |
| G09 G 5/36 | G09 G 5/36 52 O M | 5 C 0 8 2 |
| G09 G 5/37 | G09 G 5/36 52 O P | |

審査請求 未請求 請求項の数 3 書面 (全 14 頁)

| | | | |
|-----------|------------------------------|------------|---|
| (21) 出願番号 | 特願2003-373471 (P2003-373471) | (71) 出願人 | 503392253 株式会社ソフィア・クレイドル |
| (22) 出願日 | 平成15年9月25日 (2003.9.25) | (72) 発明者 | 堀口 淳史 京都府京都市左京区田中問田町2番地7 思文閣会館3F 株式会社ソフィア・クレイドル社内 |
| | | F ターム (参考) | 5B069 AA01 CA13 DD20 5C082 AA00 BB42 CA11 CA54 CA56 CA62 DA22 DA42 MM02 |

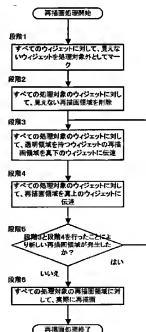
(54) 【発明の名称】 少資源環境でも高速に動作する効率的なマルチウィンドウ GUI 描画アルゴリズム

(57) 【要約】

【課題】実行速度が遅くメモリ容量も少なく自由領域型を使用できないデバイスにおいて、透過領域の描画をサポートしたマルチウィンドウ GUI を高速かつ効率的に描画する。

【解決手段】ウィジェット毎に再描画領域を四角形のみでスケジューリングし、描画処理を一つのタイミングにまとめ、効率的に描画関数を呼び出せるように可視領域と再描画領域を整理した後、各ウィジェットの描画関数を多くとも一度だけ呼び出すようにすることで透過領域を持つような複雑なマルチウィンドウ GUI を効率的に描画する。

【選択図】 図 1



【特許請求の範囲】

【請求項 1】

再描画領域を四角形型のみで管理し、それらをスケジューリングし、2つの四角形領域がスケジューリングされたときは2つの四角形領域を含む一つの最小の四角形領域として管理し、また複数の四角形領域がスケジューリングされたときはそれらすべての四角形領域を含む一つの最小の四角形領域として管理する描画方式。

【請求項 2】

ウィジェット毎に再描画領域をスケジューリングし、描画処理を一つのタイミングにまとめ、効率的に描画関数を呼び出せるように請求項3のアルゴリズムを使用し可視領域と再描画領域を整理した後、各ウィジェットの描画関数を多くとも一度だけ呼び出すことにより、透過領域を持つような複雑なマルチウィンドウ GUI を効率的に描画する方式。

10

【請求項 3】

段階1として、見えないウィジェットを処理対象外として除外し、段階2として、完全に見えない再描画領域を処理対象外として除外し、段階3として、透過領域を持ったウィジェットの再描画領域を真下に位置するウィジェットの再描画領域に追加し、段階4として、真上に位置するウィジェットに再描画領域を追加し、段階5として、段階3と段階4で新たな再描画領域が発生しなくなるまで段階3と段階4を繰り返す、段階6として、実際に再描画領域を描画する方式、または、段階1から段階6のうち一部を省略した描画方式、またはそれらの順序を入れ替えた描画方式。

【発明の詳細な説明】

20

【発明の詳細な説明】

【技術分野】

【0001】

本発明は、携帯電話端末や PDA などの少資源ハードウェア環境において、マルチウィンドウ GUI、特に透過領域を含んだマルチウィンドウ GUI を効率的に描画する方式に関するものである。

【背景技術】

【0002】

従来よりマルチウィンドウ GUI を描画する方法として以下のものが存在する。

30

1. 再描画領域を自由領域型としてスケジューリングし、自由領域型でクリッピングして描画することで、複雑なウィジェットの重なりや透過領域を処理する方式。

2. 1の応用型で、ウィジェット毎にオフスクリーンを持ちなるべく描画回数を減らす方式。

3. 再描画の必要が発生した時点で影響するすべてのウィジェットを再描画する方式。

4. 3の応用型で、ウィジェット毎にオフスクリーンを持ち、なるべく描画回数を減らす方式。

【0003】

【特許文献1】特開平9-244864号公報

【発明が解決しようとする課題】

40

【0004】

従来方法では、

【0005】

で述べた1や2のような自由領域型を使用した再描画領域の管理は、自由領域型を使用できない環境下では使用できない。また

【0006】

で述べた3や4のような頻繁に再描画処理の発生する構造はデバイスの処理速度の面から実用的ではない。更に、

【0007】

で述べた2や4の方法では描画回数をなるべく減らすという効果はあるものの、メモリを

50

多く使用するため、メモリが少ないデバイス上では実用的ではない。

【課題を解決するための手段】

【0005】

上述した課題を解決するため、本発明は、ウィジェット毎に再描画領域を四角形のみで管理し、それらをスケジューリングし、ウィジェットはオフスクリーンを保持せず、描画処理を一つのタイミングにまとめ、効率的に描画関数を呼び出せるように可視領域と再描画領域を整理した後、各ウィジェットの描画関数を多くとも一度だけ呼び出すようにする。

【発明を実施するための最良の形態】

【0006】

以下に本発明の一つの実施形態について説明するが、本発明の権利範囲はここで示される実施形態に限定されるものではない。(図4)は本発明を使用したソフトウェアのイベント処理の流れを示したものである。ユーザー操作やその他の処理によって発生したイベントは、ウィジェット1とウィジェット2で処理されるものとする。

【0007】

ウィジェット1では関数1と関数2がイベントを処理し、その結果として再描画を必要とするものとする。またウィジェット2では関数1がイベントを処理し再描画を必要とするものとする。

【0008】

本発明では再描画が必要になった時点で再描画要求領域をスケジューリングするため、ウィジェット1の関数1とウィジェット1の関数2ではウィジェット1の再描画領域に対してスケジューリングを行う(図2・図3)こととなる。またウィジェット2の関数1では、ウィジェット2の再描画領域に対してスケジューリングを行うこととなる。各ウィジェットの再描画領域は一つの四角形として管理されることとなる。

【0009】

(図1)は本発明の具体的な処理の流れを示した図である。段階1では、処理の前段階として、他のウィジェットに隠れていて見えないウィジェットや親のウィジェットの可視領域の外側にあるウィジェットを処理の対象から除外する(図5・図6)。すべてのウィジェットに対して処理を行った後、段階1は終了となる。

【0010】

次に、段階2では、他のウィジェットに隠れていて見えない再描画領域や親のウィジェットの可視領域の外側にある再描画領域など、隠れていて見えない再描画領域を処理の対象から除外する(図7・図8)。すべての処理対象のウィジェットに対して処理を行った後、段階2は終了となる。

【0011】

段階3では、透過領域を持ったウィジェットが再描画される場合、対象のウィジェットより下にあるウィジェットの再描画も必要になる(図9・図10)ため、対象の再描画領域を、真下に位置するウィジェットの再描画領域に追加する。一つ真下に位置するウィジェットが不透明かつウィジェットの領域が対象の再描画領域を完全に包含する場合、再描画領域の伝達処理はここで終了することとなる(図11)。もし、真下に位置するウィジェットが透明である場合や、再描画領域を完全に包含しない場合、更に下に位置するウィジェットにも伝達処理は行われることとなる(図12)。すべての処理対象のウィジェットに対して、下から初めて上まで処理を行った後、段階3は終了となる。

【0012】

次に、段階4では、あるウィジェットの真下のウィジェットで再描画が発生したとき、真上に位置するウィジェットも再描画が必要になる(図13・図14)ため、対象の再描画領域を、真上に位置するすべてのウィジェットの再描画領域にも追加する。このことは特に、透過領域を持ったウィジェットの状況顕著であるが、不透明なウィジェットの状況は真下のウィジェットで再描画が発生した場合、自分自身の見た目をコピーして退避させておくなどの手段で描画処理を最適化することも可能である。すべての処理対象のウィジェ

ットに対して、上から初めて下まで処理を行った後、段階4は終了となる。

【0013】

段階5では、段階3と段階4を行うことによって新たな再描画領域が発生する(図15-図18)可能性があるため、新たな再描画領域が発生しなくなるまで段階3と段階4を繰り返す処理を行う。

【0014】

段階6では、段階1から段階5ですでに再描画領域の整理が完了しているため、必要に応じて各ウィジェットの描画関数を多くとも一度だけ呼び出して、実際の描画処理を行う。すべての処理対象のウィジェットの再描画領域に対して、下から初めて上まで処理を行った後、段階6は終了となる。

【0015】

以上が本発明の具体的な処理の流れである。

【発明の効果】

【0016】

本発明によって、自由領域型を使用できない環境下や実行速度が遅くメモリ容量も少ない環境化においても、複雑なマルチウィンドウGUIの描画を効率的に行うことが可能になる。

【図面の簡単な説明】

【図1】本発明のアルゴリズム

【図2・図3】四角形型のみで再描画領域をスケジューリングして管理する方法

【図4】本発明を使用したソフトウェアのイベント処理の流れ

【図5・図6】段階1の処理の効果

【図7・図8】段階2の処理の効果

【図9・図10】段階3の処理の必要性

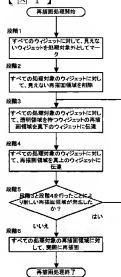
【図11】段階3の処理を打ち切れる場合

【図12】段階3の処理を打ち切れない場合

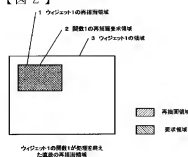
【図13・図14】段階4の処理の必要性

【図15-図16】段階3と段階4の処理での再描画領域の変遷

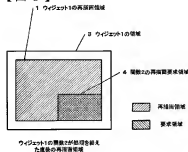
【図 1】



【図 2】



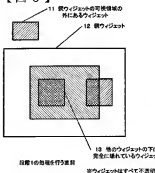
【図 3】



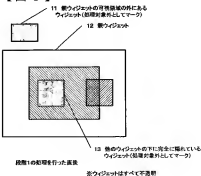
【図 4】



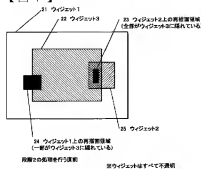
【図 5】



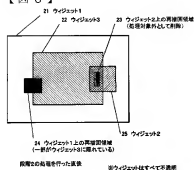
【図 6】



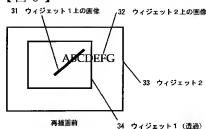
【図 7】



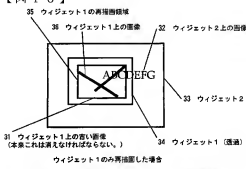
【図 8】



【図 9】

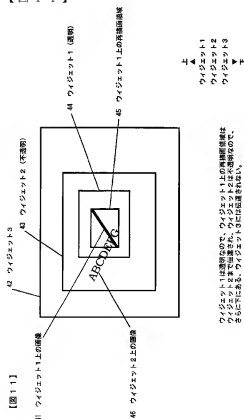


【図 10】

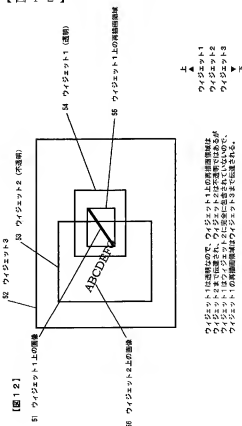


ウィジェット1は透過なので、ウィジェット1上の再描画領域を消滅させただけでは、正しい再描画画像が得られない。正解に再描画するためには、両方のウィジェット、つまりウィジェット2から再描画しなければならない。

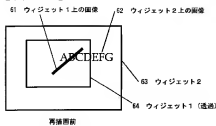
【図 11】



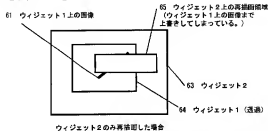
【図 12】



【図 13】

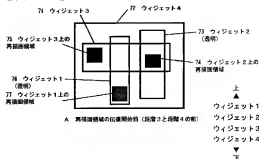


【図 14】

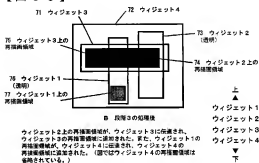


ウィジェット2は透明なので、ウィジェット2上の再描画領域を再描画しただけでは、ウィジェット1上の画像が消えてしまう。正確に再描画するためには、真上のウィジェット、つまりウィジェット1まで再描画しなければならない。

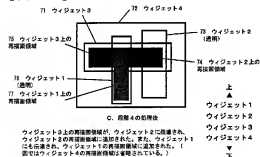
【図 15】



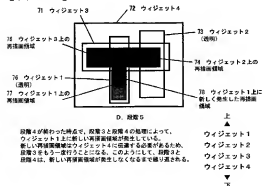
【図 16】



【図 17】



【図 18】



【手続補正書】

【提出日】平成15年12月22日(2003.12.22)

【手続補正1】

【補正対象書類名】明細書

【補正対象項目名】全文

【補正方法】変更

【補正の内容】

【発明の詳細な説明】

【技術分野】

【0001】

本発明は、携帯電話端末やPDAなどの少資源ハードウェア環境において、マルチウィンドウGUI、特に透過領域を含んだマルチウィンドウGUIを効率的に描画する方式に関するものである。

【背景技術】

【0002】

従来よりマルチウィンドウGUIを描画する方法として以下のものが存在する。

1. 再描画領域を自由領域型としてスケジューリングし、自由領域型でクリッピングして描画することで、複雑なウィジェットの重なりや透過領域を処理する方式。
2. 1の応用型で、ウィジェット毎にオフスクリーンを持ちなるべく描画回数を減らす方式。
3. 再描画の必要が発生した時点で影響するすべてのウィジェットを再描画する方式。
4. 3の応用型で、ウィジェット毎にオフスクリーンを持ち、なるべく描画回数を減らす方式。

【0003】

【特許文献1】特開平9-244864号公報

【発明が解決しようとする課題】

【0004】

従来方法では、

【0005】

で述べた1や2のような自由領域型を使用した再描画領域の管理は、自由領域型を使用できない環境下では使用できない。また

【0006】

で述べた3や4のような頻繁に再描画処理の発生する構造はデバイスの処理速度の面から実用的ではない。更に、

【0007】

で述べた2や4の方法では描画回数をなるべく減らすという効果はあるものの、メモリを多く使用するため、メモリが少ないデバイス上では実用的ではない。

【課題を解決するための手段】

【0008】

上述した課題を解決するため、本発明は、ウィジェット毎に再描画領域を四角形のみで管理し、それらをスケジューリングし、ウィジェットはオフスクリーンを保持せず、描画処理を一つのタイミングにまとめ、効率的に描画関数を呼び出せるように可視領域と再描画領域を整理した後、各ウィジェットの描画関数を多くとも一度だけ呼び出すようにする。

【発明を実施するための最良の形態】

【0009】

(図4)は本発明を使用したソフトウェアのイベント処理の流れを示したものである。ユーザー操作やその他の処理によって発生したイベントは、ウィジェット1とウィジェット2で処理されるものとする。

【0010】

ウィジェット1では関数1と関数2がイベントを処理し、その結果として再描画を必要とするものとする。またウィジェット2では関数1がイベントを処理し再描画を必要とするものとする。

【0008】

本発明では再描画が必要になった時点で再描画要求領域をスケジューリングするため、ウィジェット1の関数1とウィジェット1の関数2ではウィジェット1の再描画領域に対してスケジューリングを行う(図2・図3)こととなる。またウィジェット2の関数1では、ウィジェット2の再描画領域に対してスケジューリングを行うこととなる。各ウィジェットの再描画領域は一つの四角形として管理されることとなる。

【0009】

(図1)は本発明の具体的な処理の流れを示した図である。段階1では、処理の前段階として、他のウィジェットに隠れていて見えないウィジェットや親のウィジェットの可視領域の外側にあるウィジェットを処理の対象から除外する(図5・図6)。すべてのウィジェットに対して処理を行った後、段階1は終了となる。

【0010】

次に、段階2では、他のウィジェットに隠れていて見えない再描画領域や親のウィジェットの可視領域の外側にある再描画領域など、隠れていて見えない再描画領域を処理の対象から除外する(図7・図8)。すべての処理対象のウィジェットに対して処理を行った後、段階2は終了となる。

【0011】

段階3では、透過領域を持ったウィジェットが再描画される場合、対象のウィジェットより下にあるウィジェットの再描画も必要になる(図9・図10)ため、対象の再描画領域を、真下に位置するウィジェットの再描画領域に追加する。一つ真下に位置するウィジェットが不透明かつウィジェットの領域が対象の再描画領域を完全に包含する場合、再描画領域の伝達処理はここで終了することとなる(図11)。もし、真下に位置するウィジェットが透明である場合や、再描画領域を完全に包含しない場合、更に下に位置するウィジェットにも伝達処理は行われることとなる(図12)。すべての処理対象のウィジェットに対して、下から初めて上まで処理を行った後、段階3は終了となる。

【0012】

次に、段階4では、あるウィジェットの真下のウィジェットで再描画が発生したとき、真上に位置するウィジェットも再描画が必要になる(図13・図14)ため、対象の再描画領域を、真上に位置するすべてのウィジェットの再描画領域にも追加する。このことは特に、透過領域を持ったウィジェットの場面顕著であるが、不透明なウィジェットの場合は真下のウィジェットで再描画が発生した場合、自分自身の見た目をコピーして退避させておくなどの手段で描画処理を最適化することも可能である。すべての処理対象のウィジェットに対して、上から初めて下まで処理を行った後、段階4は終了となる。

【0013】

段階5では、段階3と段階4を行うことによって新たな再描画領域が発生する(図15・図18)可能性があるため、新たな再描画領域が発生しなくなるまで段階3と段階4を繰り返す処理を行う。

【0014】

段階6では、段階1から段階5ですでに再描画領域の整理が完了しているため、必要に応じて各ウィジェットの描画関数を多くとも一度だけ呼び出して、実際の描画処理を行う。すべての処理対象のウィジェットの再描画領域に対して、下から初めて上まで処理を行った後、段階6は終了となる。

【0015】

以上が本発明の具体的な処理の流れである。

【発明の効果】

【0016】

本発明によって、自由領域型を使用できない環境下や実行速度が遅くメモリ容量も少ない

環境化においても、複雑なマルチウィンドウGUIの描画を効率的に行うことが可能になる。

【図面の簡単な説明】

【図1】本発明のアルゴリズム

【図2】四角形形のみで再描画領域をスケジューリングして管理する方法（処理前）

【図3】四角形形のみで再描画領域をスケジューリングして管理する方法（処理後）

【図4】本発明を使用したソフトウェアのイベント処理の流れ

【図5】段階1の処理の効果（処理前）

【図6】段階1の処理の効果（処理後）

【図7】段階2の処理の効果（処理前）

【図8】段階2の処理の効果（処理後）

【図9】段階3の処理の必要性（仮定図）

【図10】段階3の処理の必要性（予測図）

【図11】段階3の処理を打ち切れる場合

【図12】段階3の処理を打ち切れない場合

【図13】段階4の処理の必要性（仮定図）

【図14】段階4の処理の必要性（予測図）

【図15】段階3の処理での再描画領域の変遷（処理前）

【図16】段階3の処理での再描画領域の変遷（処理後）

【図17】段階4の処理での再描画領域の変遷（処理後）

【図18】段階5の処理が必要な状況

【符号の説明】

- 1 ウィジェット1の再描画領域
- 2 関数1の再描画要求領域
- 3 ウィジェット1の領域
- 4 関数2の再描画要求領域
- 5 親ウィジェットの可視領域の外にあるウィジェット
- 6 親ウィジェット
- 7 他のウィジェットの下に完全に隠れているウィジェット
- 8 ウィジェット1
- 9 ウィジェット3
- 10 ウィジェット2上の再描画領域
- 11 ウィジェット1上の再描画領域
- 12 ウィジェット2
- 13 ウィジェット1上の画像
- 14 ウィジェット2上の画像
- 15 ウィジェット2
- 16 ウィジェット1
- 17 ウィジェット1の再描画領域
- 18 ウィジェット1上の新しい画像
- 19 ウィジェット1上の画像
- 20 ウィジェット3
- 21 ウィジェット2
- 22 ウィジェット1
- 23 ウィジェット1上の再描画領域
- 24 ウィジェット2上の画像
- 25 ウィジェット4
- 26 ウィジェット4上の画像
- 27 ウィジェット1上の画像
- 28 ウィジェット2上の画像

- 29 ウィジェット2
- 30 ウィジェット1
- 31 ウィジェット2上の再描画領域
- 32 ウィジェット3
- 33 ウィジェット4
- 34 ウィジェット2
- 35 ウィジェット2上の再描画領域
- 36 ウィジェット3上の再描画領域
- 37 ウィジェット1
- 38 ウィジェット1上の再描画領域
- 39 ウィジェット1上に新しく発生した再描画領域

【手続補正2】

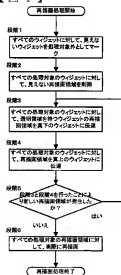
【補正対象書類名】図面

【補正対象項目名】全図

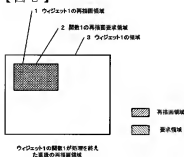
【補正方法】変更

【補正の内容】

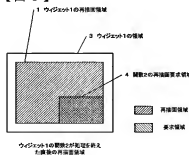
【図1】

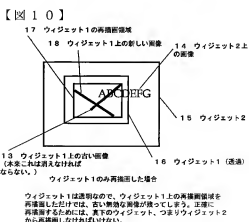
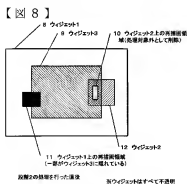
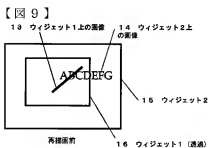
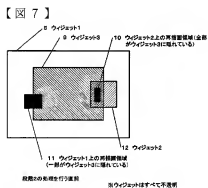
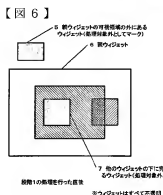
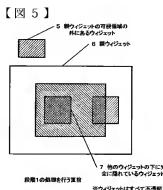


【図2】

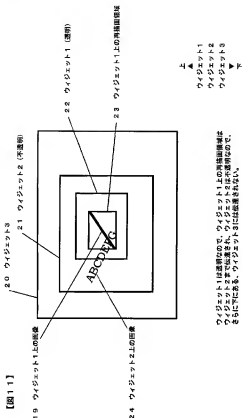


【図3】

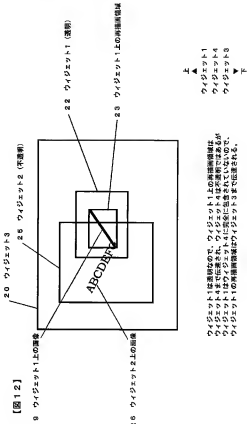




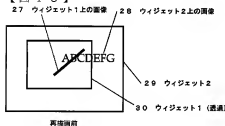
【 例 1 1 】



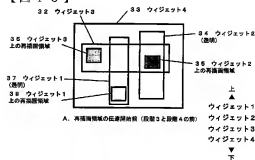
【图 1 2】



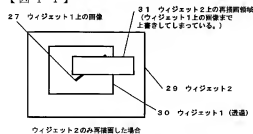
【图 13】



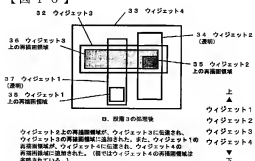
【图 15】



【 例 1 4 】



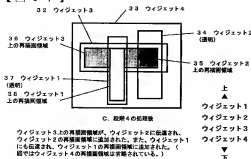
【图 16】



ウィジェット1は透明なので、ウィジェット2上の再描画領域を再描画しただけでは、ウィジェット1上の画像が消えてしまう。正確に再描画するためには、真上のウィジェット、つまりウィジェット1まで再描画しなくてはならない。

ウィジェット2上の再描画領域が、ウィジェット3に伝達され、ウィジェット3の再描画領域に追加された。また、ウィジェット1の再描画領域が、ウィジェット4に伝達され、ウィジェット4の再描画領域に追加された。(但しではウィジェット4の再描画領域は空である。)

【図17】



【図18】

